01-06-06

AF IZW

PTO/SB/21 (09-04)
Approved for use through 07/31/2006. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

TRANSMITTAL FORM

(to be used for all correspondence after initial filing)

Application Number	09/880,632-Conf. #5911
Filing Date	June 12, 2001
First Named Inventor	Wenting Tang
Art Unit	2157
Examiner Name	G. G. Todd
Attorney Docket Number	10012351-1
Total Number of Pages in This Submission	41

ENCLOSURES (*Check all that apply*)

- [ ] Fee Transmittal Form
  - [ ] Fee Attached
- [ ] Amendment/Reply
  - [ ] After Final
  - [ ] Affidavits/declaration(s)
- [ ] Extension of Time Request
- [ ] Express Abandonment Request
- [ ] Information Disclosure Statement
- [ ] Certified Copy of Priority Document(s)
- [ ] Reply to Missing Parts/ Incomplete Application
  - [ ] Reply to Missing Parts under 37 CFR 1.52 or 1.53
- [ ] Drawing(s)
- [ ] Licensing-related Papers
- [ ] Petition
- [ ] Petition to Convert to a Provisional Application
- [ ] Power of Attorney, Revocation Change of Correspondence Address
- [ ] Terminal Disclaimer
- [ ] Request for Refund
- [ ] CD, Number of CD(s) ______
  - [ ] Landscape Table on CD
- [ ] After Allowance Communication to TC
- [ ] Appeal Communication to Board of Appeals and Interferences
- [x] Appeal Communication to TC **(Appeal Notice, Brief, Reply Brief)**
- [ ] Proprietary Information
- [ ] Status Letter
- [x] Other Enclosure(s) (please Identify below):

2 Return Postcards
Certificate of Mail

Remarks

SIGNATURE OF APPLICANT, ATTORNEY, OR AGENT

Firm Name	FULBRIGHT & JAWORSKI L.L.P., Attorney, Agent for Applicant		
Signature			
Printed name	Jody C. Bishop		
Date	January 5, 2006	Reg. No.	44,034

HEWLETT-PACKARD COMPANY
Intellectual Property Administration
P.O. Box 272400
Fort Collins, Colorado 80527-2400

Docket No.: 10012351-1
(PATENT)

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent Application of:
Wenting Tang et al.

Application No.: 09/880,632

Confirmation No.: 5911

Filed: June 12, 2001

Art Unit: 2157

For: METHOD AND SYSTEM FOR A FRONT-END MODULAR TRANSMISSION CONTROL PROTOCOL (TCP) HANDOFF DESIGN IN A STREAMS BASED TRANSMISSION CONTROL PROTOCOL INTERNET PROTOCOL (TCP/IP) IMPLEMENTATION

Examiner: G. G. Todd

APPEAL BRIEF

MS Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir:

In response to the Notification of Non-Compliant Appeal Brief mailed December 21, 2005, Appellant hereby submits this Appeal Brief that corrects the identified deficiencies of the Appeal Brief filed September 19, 2005. Specifically, this Appeal Brief identifies a related appeal in section II hereof and includes Appendices B and C. Otherwise, this Appeal Brief is identical to the Appeal Brief filed September 19, 2005. Thus, this Appeal Brief is in furtherance of the Notice of Appeal filed on July 18, 2005 and is believed to be in full compliance with 37 C.F.R. § 41.37.

The fees required under § 41.20(b)(2) were dealt with in the Appeal Brief filed September 19, 2005. No further fees are believed to be due for this modified brief in response to the Notification of Non-Compliant Appeal Brief of December 21, 2005.

This brief contains items under the following headings as required by 37 C.F.R. § 41.37 and M.P.E.P. § 1206:

I. REAL PARTY IN INTEREST

The real party in interest for this appeal is:

Hewlett-Packard Development Company, L.P., a Texas Limited Partnership having its principal place of business in Houston, Texas.

II. RELATED APPEALS, INTERFERENCES, AND JUDICIAL PROCEEDINGS

Appellant did not identify any related appeals in the Appeal Brief filed September 19, 2005. Because the present application does not reference co-pending U.S. Patent Application Serial No. 09/880,631, Appellant did not consider such co-pending application as related. However, because the Examiner asserts in the Notification of Non-Compliant Appeal Brief of December 21, 2005 that the Appeal Brief should identify the appeal of co-pending U.S. Patent Application Serial No. 09/880,631 as a related appeal, Appellant hereby does so.

A notice of appeal was filed for co-pending U.S. Patent Application Serial No. 09/880,631 (hereafter "the '631 application") on July 19, 2005. The claims of the '631 application are rejected on the same grounds as the claims of the present application, i.e., as being anticipated by U.S. Patent No. 6,775,692 issued to Albert et al ("*Albert*"). Thus, the appeal of the '631 application (and particularly the Board's interpretation of *Albert*) may affect, be affected by, or have a bearing on the Board's decision in this appeal.

There are no other appeals, interferences, or judicial proceedings which will directly affect or be directly affected by or have a bearing on the Board's decision in this appeal.

III. STATUS OF CLAIMS

A. Total Number of Claims in Application

There are 29 claims pending in application.

B. Current Status of Claims

1. Claims canceled: None
2. Claims withdrawn from consideration but not canceled: None
3. Claims pending: 1-29
4. Claims allowed: None
5. Claims rejected: 1-29

C. Claims On Appeal

The claims on appeal are claims 1-29

IV. STATUS OF AMENDMENTS

A first Office Action was mailed for this application September 22, 2004. In response, Applicant filed an Amendment on December 22, 2004, which presented amendments to claims 3, 5, 17, and 22. A Final Office Action was then mailed April 20, 2005. Applicant did not file an amendment in response to the Final Office Action, but instead filed a Notice of Appeal, which this brief supports. Thus, the claims on appeal are those claims rejected in the Final Office Action, and a listing of those claims are provided in Appendix A hereto.

V. SUMMARY OF CLAIMED SUBJECT MATTER

The following provides a concise explanation of the subject matter defined in the claims involved in the appeal, referring to the specification by page and line number and to the drawings by reference characters, as required by 37 C.F.R. § 41.37(c)(1)(v). Each element of the claims is identified by a corresponding reference to the specification and drawings where applicable. Note that the citation to passages in the specification and drawings for each claim element does not imply that the limitations from the specification and drawings should be read into the corresponding claim element.

According to one claimed embodiment of the present invention, a method of TCP state migration in a communication network comprises establishing a communication session between a client (client 210 of Figure 2) and a front-end node (front-end node 252 of Figure 2) at a first bottom TCP (BTCP) module (BTCP module 350 of Figures 3C and 5, and *see* page 7, lines 16-29 and page 24, lines 13-25 of the specification), located below a first TCP module (TCP module 320 of Figures 3B, 3C and 5, and *see* page 7, lines 16-24 and page 19, line 28 through page 20, line 27 of the specification) in a first operating system at the front-end node. The front-end node accesses a plurality of back-end web servers (back-end web servers 255, 257, and 259 of Figure 2, and *see* page 7, lines 5-15 of the specification) forming a web server cluster that contains content. The method further comprises receiving a HTTP request from the client at the first BTCP module (block 760 of Figure 7, and *see* page 8, lines 1-3 and page 28, lines 24-28 of the specification), and parsing the HTTP request to determine which back-end web server ("a selected back-end web server") in the plurality of back-end web servers can process the HTTP request (block 620 of Figure 6, and *see* page 8, lines 1-10, page 26, lines 4-10, and page 28, lines 26-28 of the specification). The selected back-end web server is not the front-end node. The method further comprises extending the communication session to the selected back-end web server by handing-off an initial TCP state of the first BTCP module to the selected back-end web server (block 630 of Figure 6, and *see* page 8, lines 11-15, page 8, line 23 through page 9, line 2, and page 26, lines 12-23 of the specification), and sending the HTTP request to the selected back-end web server. The method further comprises switching a bottom IP (BIP) module at the front-end node to a forwarding mode, wherein packets received at the BIP module from the client are forwarded to the selected back-end web server (block 640 of Figure 6, and *see* page 8, lines 17-21 and

page 26, lines 25-29 of the specification). The BIP module (BIP module 360 of Figures 3C and 5, and *see* page 24, lines 13-25 of the specification) is located below an IP module (IP module 330 of Figures 3B, 3C and 5, and *see* page 19, line 28 through page 20, line 27 of the specification) at the front-end node. The method further comprises terminating the communication session at the front-end node after the HTTP request is fully processed (*see* page 27, lines 1-5 and page 33, lines 8-16 of the specification).

In one embodiment, the back-end web server includes a second BTCP module (BTCP module 520 of Figure 5) that is located below a second TCP module (TCP module 530 of Figure 5) in a second operating system.

In one embodiment, extending the communication session to the selected back-end web server by handing-off an initial TCP state further comprises sending a SYN packet to said selected back-end web server (block 810 of Figure 8). The SYN packet is intercepted by a second BTCP module (BTCP module 520 of Figure 5, and block 830 of Figure 8). The SYN packet is originally sent from the client to the front-end node in requesting the communication session. The SYN packet is stored at the first BTCP module. The extending further comprises including an initial sequence number within the SYN packet that enables the second BTCP module to understand a proper TCP state of the first BTCP module in the communication session (block 820 of Figure 8). The extending further comprises receiving a SYN/ACK packet from the selected back-end web server (block 850 of Figure 8), the SYN/ACK packet updated by the second BTCP module to reflect the proper TCP state of the first BTCP module (block 860 of Figure 8). And, the extending further comprises sending an ACK packet from the first BTCP module to the selected back-end web server (block 880 of Figure 8), the ACK packet originally sent from the client to the front-end node in establishing the communication session.

In one embodiment, the method further comprises sending response packets from the selected back-end web server to the client in a communication path that does not include the front-end node by changing headers of the response packets such that it appears that the source of said response packets is the first BTCP in its proper TCP state (*see* Figure 2, and *see* page 17, lines 5-15 of the specification).

In one embodiment, terminating the communication session further comprises intercepting TCP control packets from a second TCP module (TCP module 530 of Figure 5) located at the selected back-end web server at the second BTCP module (*see* page 33, lines 8-11 of the specification); sending the TCP control packets to the first BTCP module from the second BTCP module (page 33, lines 8-11 of the specification); sending the TCP control packets to the client from the first BTCP module (page 33, lines 11-14 of the specification); and terminating the communication session at the front-end node and the back-end web server (*see* page 9, lines 4-15 of the specification).

According to another claimed embodiment, a method of TCP state migration in a communication network comprises receiving a request from a client (client 210 of Figure 2) for establishing a communication session at a first bottom TCP (BTCP) module (BTCP module 350 of Figures 3C and 5, and *see* page 7, lines 16-29 and page 24, lines 13-25 of the specification) located at a front-end node (front-end node 252 of Figure 2). The front-end node accesses a plurality of back-end web servers containing content (back-end web servers 255, 257, and 259 of Figure 2, and *see* page 7, lines 5-15 of the specification), wherein the content is partially replicated between each of the plurality of back-end web servers. The communication session is established for the transfer of data contained within the content to the client. The method further comprises establishing the communication session between the client and the first BTCP module. The first BTCP module is located below a first TCP module (TCP module 320 of Figures 3B, 3C, and 5, and *see* page 7, lines 16-24 and page 19, line 28 through page 20, lines 27 of the specification) in a first operating system at the front-end node. The method further comprises receiving a HTTP request from the client at the first BTCP module (block 760 of Figure 7, and *see* page 8, lines 1-3 and page 28, lines 24-28 of the specification), and parsing the HTTP request to determine which back-end web server ("a selected back-end web server") in the plurality of back-end web servers contains the data in order to process the HTTP request (block 620 of Figure 6, and *see* page 8, lines 1-10, page 26, lines 4-10, and page 28, lines 26-28 of the specification). The selected back-end web server is not the front-end node. The method further comprises extending the communication session to the selected back-end web server by handing-off an initial TCP state of the first BTCP module to a second BTCP module located at the selected back-end web server (block 630 of Figure 6, and *see* page 8, lines 11-15, page 8, lines 23 through page 9, line 2, and page 26, lines 12-23 of the specification). The initial TCP state is associated with the

communication session between the client and the first BTCP module, and the second BTCP module (BTCP module 520 of Figure 5) is located below a second TCP module (TCP module 530 of Figure 5) in a second operating system at the selected back-end web server. The method further comprises sending the HTTP request to the selected back-end web server. The method further comprises switching a bottom IP (BIP) module in the front-end node to a forwarding mode, wherein packets, from the client, received at the front-end node are intercepted by the BIP module and forwarded to the selected back-end web server (block 640 of Figure 6, and *see* page 8, lines 17-21 and page 26, lines 25-29 of the specification). The BIP module (BIP module 360 of Figures 3C and 5, and *see* page 24, lines 13-25 of the specification) is located below an IP module (IP module 330 of Figures 3C, 3C, and 5, and *see* page 19, line 28 through page 20, line 27 of the specification) in the front-end node. The BIP module changes the destination IP addresses of the packets to the selected back-end web server (page 32, line 10 through page 33, line 6 of the specification). The method further comprises terminating the communication session after the HTTP request has been fully processed (page 26, lines 1-5 and page 33, lines 8-16 of the specification).

In one embodiment, extending the communication session further comprises storing a SYN packet sent from the client to the front-end node, the SYN packet requesting the communication session (page 9, lines 16-19 of the specification); storing an ACK packet sent from the client to the front end node in establishing the communication session (page 9, lines 19-22 of the specification); sending the SYN packet to the selected back-end web server so that it appears that the SYN packet originated from the client (page 9, lines 24-29 of the specification); sending the initial TCP state to the second BTCP module, including the initial sequence number, that enables the second BTCP module to understand a proper TCP state of the first BTCP module for the communication session (page 10, lines 1-10 of the specification); receiving a SYN/ACK packet at the first BTCP module from the second TCP module, the SYN/ACK packet updated by the second BTCP module to reflect the proper TCP state at the first BTCP for the communication session (page 10, lines 11-13 of the specification); and sending the ACK packet to the selected back-end web server to extend the communication session to the selected server (page 10, lines 16-21 of the specification).

In one embodiment, the method further comprises sending response packets from the back-end web server to the client in a communication path that does not include the front-end

node, by changing headers of the response packets such that it appears that the source of the response packets is the front-end node with the proper TCP state (*see* Figure 2, and *see* page 17, lines 5-15 of the specification).

In one embodiment, terminating the communication session further comprises intercepting TCP control packets from the selected back-end web server at the second BTCP module (*see* page 33, lines 8-11 of the specification); sending the TCP control packets to the first BTCP module from the second BTCP module (page 33, lines 8-11 of the specification); sending the TCP control packets to the client from the first BTCP module (page 33, lines 11-14 of the specification); and terminating the communication session at the front-end node and the back-end web server (*see* page 9, lines 4-15 of the specification).

In one embodiment, the method bypasses the first TCP module (*see* Figure 5, where TCP module 320 is bypassed).

According to another claimed embodiment, a communication network for TCP state migration comprises a client (client 210 of Figure 2) and a front-end node (front-end node 252 of Figure 2) coupled to the client by the communication network. The front-end node includes a front-end bottom TCP (BTCP) module (BTCP module 350 of Figures 3C and 5, and *see* page 7, lines 16-29 and page 24, lines 13-25 of the specification) located below a front-end TCP module (TCP module 320 of Figures 3B, 3C, and 5, and *see* page 7, lines 16-24 and page 19, lines 28 through page 20, line 27 of the specification) in a first operating system, and a bottom IP (BIP) module (BIP module 360 of Figures 3C and 5, and *see* page 24, lines 13-25 of the specification) located below an IP module (IP module 330 of Figures 3B, 3C, and 5, and see page 19, line 28 through page 20, lines 27 of the specification) in the first operating system. The communication network further comprises a plurality of back-end web servers (back-end web servers 255, 257, and 259 of Figure 2, and *see* page 7, lines 5-15 of the specification) including a selected back-end web server. The plurality of back-end web servers contain content that is partitioned between each of the plurality of back-end web servers. Each of the plurality of back-end web servers is coupled to the front-end node through the communication network. Each of the plurality of back-end web servers includes a back-end bottom TCP module (BTCP module 520 of Figure 5) located below a back-end TCP module (TCP module 530 of Figure 5).

In one embodiment, the front-end BTCP module establishes a communication session with the client for the transfer of data contained within the content to the client (page 7, lines 16-29 of the specification).

In one embodiment, the front-end BTCP module parses a HTTP request from the client in order to determine which of the plurality of back-end web servers, a selected back-end web server, contains the data in order to process the HTTP request (page 8, lines 1-10 of the specification).

In one embodiment, the front-end BTCP module extends the communication session to the selected back-end web server by handing-off an initial TCP state of the front-end BTCP module to a second BTCP module (BTCP module 520 of Figure 5) located at the selected back-end web server (page 8, lines 11-15 of the specification). The initial TCP state is associated with a proper TCP state for the front-end BTCP module in the communication session. The front-end BTCP module further forwards packets, including the HTTP request, from the client after successfully handing-off the initial TCP state (page 8, line 16 – page 9, line 2 of the specification).

In one embodiment, the second BTCP module understands the proper TCP state of the front-end BTCP module in the communication session and modifies headers in response packets from the selected back-end web server to reflect the proper TCP state (page 10, lines 1-10 of the specification).

In one embodiment, the BIP module changes a destination address in forwarding the packets from the client (page 32, lines 10 – page 33, line 6 of the specification).

In one embodiment, the second BTCP module located at the selected back-end web server sends the response packets from the selected back-end web server to the client in a communication path that does not include the front-end node by changing headers of the response packets such that it appears the source of the response packets is the front-end node (*see* Figure 2, and *see* page 17, lines 5-15 of the specification).

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Claims 1-29 are rejected under 35 U.S.C. § 102(e) as being anticipated by U.S. Patent No. 6,775,692 issued to Albert et al (hereinafter "*Albert*").

VII. ARGUMENT

Appellant respectfully traverses the outstanding rejections of the pending claims, and requests that the Board reverse the outstanding rejections in light of the remarks contained herein. Below, Appellant argues many of the rejected claims separately. Thus, Appellant respectfully asserts that separately argued claims do not stand or fall together, *see* 37 C.F.R. § 41.37(c)(1)(vii).

To anticipate a claim under 35 U.S.C. § 102, a single reference must teach every element of the claim, *see* M.P.E.P. § 2131. Appellant respectfully submits that *Albert* fails to teach each and every element of claims 1-37, as discussed further below.

<u>Independent Claim 1</u>

Independent claim 1 recites:

> In a communication network, a method of TCP state migration comprising the steps of:
> a) establishing a communication session between a client and a front-end node <u>at a first bottom TCP (BTCP) module located below a first TCP module in a first operating system at said front-end node</u>, said front-end node accessing a plurality of back-end web servers forming a web server cluster that contains content;
> b) receiving a HTTP request from said client at said first BTCP module;
> c) <u>parsing said HTTP request to determine which back-end web server, a selected back-end web server, in said plurality of back-end web servers can process said HTTP request</u>, said selected back-end web server not said front-end node;
> d) extending said communication session to said selected back-end web server by <u>handing-off an initial TCP state of said first BTCP module to said selected back-end web server</u>;
> e) sending said HTTP request to said selected back-end web server;
> f) <u>switching a bottom IP (BIP) module at said front-end node to a forwarding mode</u>, wherein packets received at said BIP module from said client are forwarded to said selected back-end web server, <u>said BIP module</u>

located below an IP module at said front-end node; and

g) terminating said communication session at said front-end node after said HTTP request is fully processed. (Emphasis added).

Albert fails to teach all elements of independent claim 1. As described further below, *Albert* does not provide a modularized solution, and thus fails to teach at least the modules recited in claim 1. Accordingly, without conceding that *Albert* teaches any of the other elements of claim 1, *Albert* fails to teach at least those elements described further below.

Specifically, *Albert* fails to teach at least:

A) a first bottom TCP (BTCP) module located below a first TCP module in a first operating system at a front-end node;

B) parsing an HTTP request to determine which back-end web server can process the HTTP request;

C) a bottom IP (BIP) module located below an IP module at the front-end node; and

D) handing-off an initial TCP state of said first BTCP module to said selected back-end web server.

A. Albert fails to teach a first bottom TCP (BTCP) module located below a first TCP module in a first operating system at a front-end node, as recited by claim 1

First, *Albert* fails to teach a first bottom TCP (BTCP) module located below a first TCP module in a first operating system at a front-end node, as recited by claim 1.

The Final Office Action cites col. 7, lines 36-60 of *Albert* in support of its assertion that *Albert* teaches this element of claim 1, *see* pages 2-3 of the Final Office Action. Col. 7, lines 36-60 of *Albert* provides:

> FIG. 2A is a block diagram of a network architecture that provides network services without requiring a network service appliance to be physically placed at a node through which all incoming and outgoing packets processed by a group of servers must pass. Several clients 201, 202, and 203 are connected to a network 210. Network 210 is connected to a group of servers 220 that includes servers 221, 222, and 223. There is no point through which all traffic between devices connected to network 210 and the group of servers 220 must pass. Instead, some traffic from network 210 that is bound for the group of servers passes through a forwarding agent 231 and some traffic between network 210 and group of servers 220 passes though a forwarding agent 232.

> In the example shown, forwarding agent 231 is connected to server 221 and server 222 and forwarding agent 232 is connected to server 222 and server 223. Thus, server 222 may communicate with network 210 through either of the forwarding agents, server 221 communicates with network 210 exclusively through forwarding agent 231, and server 223 communicates with network 210 exclusively through forwarding agent 232. This arrangement may be generalized to include an arbitrary number of servers connected to an arbitrary number of forwarding agents with individual servers connected to arbitrary subsets of the forwarding agents.

The above portion of *Albert* teaches forwarding agents 231 and 232 through which traffic flows between the clients and the servers. The Office Action appears to contend that the forwarding agents of *Albert* are front-end nodes, as recited by claim 1. However, *Albert* in no way teaches that either of the forwarding agents 231 and 232 include a first bottom TCP (BTCP) module located below a first TCP module in a first operating system. *Albert* does not teach a module at all in the above relied-upon portion, and certainly not a BTCP module located below a TCP module in an operating system.

In response to the above argument, the Final Office Action further asserts at page 10 thereof:

> A module can be defined as "A self-contained functional unit which is used with a larger system. A software module is a part of a program that performs a particular task. A hardware module can be a packaged unit that attaches to a system." Albert is teaching a forwarding agent having an interface (at least Fig. 2C) and connected to a plurality of back-end nodes or servers (at least Fig. 2A) through which communication with a client is allowed, thus a front-end module or program performing a particular task directing client requests to servers and offering SYN ACK packets (BTCP) and TCP flow (at least col. 13, lines 10-29; col. 8, lines 17-39).

The above-portion of the Final Office Action appears to quote a definition for "module," but fails to identify the source of the definition. Further, the above-quoted portion of the Final Office Action appears to assert that the forwarding agent of *Albert* has a BTCP module. However, *Albert* simply does not teach that its forwarding agent includes a BTCP module as recited by claim 1. While *Albert* teaches a forwarding agent that is capable of receiving requests and forwarding them to back-end nodes, it does <u>not</u> teach that the forwarding agent and back-end nodes comprise TCP handoff modules. *Albert* simply provides no teaching of a modularized solution.

B. Albert fails to teach parsing an HTTP request to determine which back-end web server can process the HTTP request, as recited by claim 1

Additionally, independent claim 1 further recites "b) receiving a HTTP request from said client at said first BTCP module; c) <u>parsing said HTTP request to determine which back-end web server, a selected back-end web server, in said plurality of back-end web servers can process said HTTP request</u>, said selected back-end web server not said front-end node" (emphasis added). *Albert* fails to teach parsing a received HTTP request to determine which of a plurality of back-end web servers can process the HTTP request.

The Final Office Action cites col. 9, lines 10-34 and 45-58 of *Albert* in support of its assertion that *Albert* teaches this element of claim 1, *see* page 3 of the Final Office Action. Col. 9, lines 10-34 and 45-58 of *Albert* provides:

> In addition to specifying instructions for each flow, service managers must also obtain information about each new flow from the forwarding agents. For example, when a service manager provides load balancing through a set of forwarding agents, the service manager uses fixed affinities to provide specific instructions to the forwarding agents detailing where packets for each load balanced flow are to be forwarded. In addition to providing those specific instructions, the service manager also provides general instructions to each forwarding agent that specify which new flows the service manager is interested in seeing. These general instructions are provided using wildcard affinities. Wildcard affinities, which are described in detail below, specify sets of flows that are of interest to a service manager. In one embodiment, this is done by specifying subnet masks that determine sets of source and destination IP addresses that will be forwarded to a service manager. In addition, ports or sets of ports and protocol may be specified in wildcard affinity as well. As is described further below, the use of wildcard affinities enables separate service managers to be configured to provide services for different sets of flows. Each service manager specifies the flows of interest to it and other service managers handle other flows. In this manner, service managers can be configured in parallel to share load.
>
> * * *
>
> In the case of load balancing, service managers send wildcard affinities to forwarding agents. The wildcard affinities specify destination IP addresses that correspond to virtual IP addresses of server clusters that are to be load balanced by the service manager. The forwarding agents then forward new packets sent to those virtual IP addresses to the appropriate service manager. The service manager selects a server from the server cluster and then the service manager sends a fixed affinity to each forwarding agent that instructs the forwarding agent to forward packets for that specific flow to the selected server in the cluster. Forwarding agents may also forward packets for purposes

other than load balancing. Packets may be forwarded to real IP addresses as well as virtual IP addresses.

Albert does not teach parsing a received HTTP request to determine which back-end web server can process such HTTP request. Rather, *Albert* teaches pre-setting a wildcard affinity based on a client's IP address. For instance, the above portion of *Albert* teaches that wildcard affinities specify sets of flows that are of interest to a service manager. The wildcard affinities are pre-selected (e.g., before even receiving a request from a client), to specify a subnet mask that identifies a set of source and destination IP addresses. Thus, for instance, a particular IP address corresponding to a client of interest can be identified by a wildcard affinity.

As described further in *Albert* at col. 12, line 6 – col. 14, line 15, a SYN packet is used to identify whether the flow matches a wildcard affinity, in which case it is forwarded to the service manager for determination of how to handle the flow. Thus, the service manager in *Albert* selects a back-end server responsive to receipt of a SYN packet, which is an initial connection establishment packet sent before it is even known what the request will be. That is, the SYN packet upon which *Albert* selects a server, does <u>not</u> include an HTTP request. Rather, the SYN packet includes source and destination IP addressed, from which it is determined by the forwarding agents whether such packet corresponds to a pre-set wildcard affinity. Thus, the back-end server is not selected in *Albert* as a result of parsing a received HTTP request, as the back-end server is selected based on other information (e.g., source IP address) included in the SYN packet.

C. Albert does not teach a bottom IP (BIP) module located below an IP module at the front-end node, as recited by claim 1

Further, claim 1 recites "switching a bottom IP (BIP) module at said front-end node to a forwarding mode, wherein packets received at said BIP module from said client are forwarded to said selected back-end web server, <u>said BIP module located below an IP module at said front-end node</u>" (emphasis added). As described above, *Albert* does not teach modules. Specifically, *Albert* does not teach a BIP module located below an IP module at the front-end node (e.g., at the forwarding agent of *Albert*).

The Final Office Action cites col. 14, lines 1-15 of *Albert* in support of its assertion that *Albert* teaches this element of claim 1, *see* page 3 of the Final Office Action. Col. 14, lines 1-15 of *Albert* provides:

> Client 304 sends a data packet to forwarding agent 302. Forwarding agent 302 has stored the fixed affinity corresponding to the flow from the client to the host in a fixed affinity database 303. Forwarding agent 302 notes the match of the 5-tuple of the data packet with an affinity key in the fixed affinity database and then forwards the data packet according to the action defined in that fixed affinity. In this example, the action defined is to translate the destination IP address of the client from the virtual IP address of virtual machine 310 to the IP address of host 306. In addition to forwarding the data packet, the affinity found by the forwarding agent also includes an action that requires the forwarding agent to send an affinity packet to service manager 300 that includes data about the packet for the purpose of service manager 300 gathering statistics about network traffic.

The above portion of *Albert* does not teach a module, and particularly not a BIP module located below an IP module at said front-end node, as recited by claim 1. Instead, the above portion of *Albert* simply teaches that the client sends a data packet to the forwarding agent, and the forwarding agent translates (using the affinity key) the destination IP address of the client from the virtual IP address of the virtual machine to the IP address of a specific host. The forwarding agent also sends an affinity packet to the service manager so that the service manager can gather statistics about network traffic. There is no teaching or even a hint in *Albert* of a BIP module located below an IP module at the front-end node (e.g., the forwarding agent). The forwarding actions performed by *Albert*, including the forwarding agent accessing the affinity database to determine a forwarding action based on a match in the database with the affinity key, are simply not taught as being performed via modules. Thus, the solution of *Albert* suffers at least from the drawbacks identified in the present application as associated with non-modularized implementations.

D. Albert fails to teach handing-off an initial TCP state of said first BTCP module to said selected back-end web server as recited by claim 1

Further still, claim 1 recites "d) extending said communication session to said selected back-end web server by <u>handing-off an initial TCP state of said first BTCP module to said selected back-end web server</u>" (emphasis added). *Albert* fails to teach this further element of claim 1. Instead, the forwarding agent in *Albert* acts as an intermediary to forward

communication that it receives to an appropriate back-end server, but the forwarding agent does not hand-off an initial TCP state to a selected back-end server. That is, the forwarding agent forwards packets to the appropriate back-end server, but it does not hand off an initial TCP state to the back-end server.

Accordingly, *Albert* fails to teach at least the above-identified elements of claim 1, and therefore claim 1 is not anticipated under 35 U.S.C. § 102 by *Albert*. As such, Appellant respectfully requests that the rejection of claim 1 be overturned.

Independent Claim 11

Independent claim 11 recites:

> In a communication network, a method of TCP state migration comprising the steps of:
>
> a) receiving a request from a client for establishing a communication session at a first bottom TCP (BTCP) module located at a front-end node, said front-end node accessing a plurality of back-end web servers containing content, wherein said content is partially replicated between each of said plurality of back-end web servers, said communication session established for the transfer of data contained within said content to said client;
>
> b) establishing said communication session between said client and said first BTCP module, <u>said first BTCP module located below a first TCP module in a first operating system at said front-end node;</u>
>
> c) receiving a HTTP request from said client at said first BTCP module;
>
> d) <u>parsing said HTTP request to determine which back-end web server, a selected back-end web server, in said plurality of back-end web servers contains said data in order to process said HTTP request</u>, said selected back-end web server not said front-end node;
>
> e) extending said communication session to said selected back-end web server by <u>handing-off an initial TCP state of said first BTCP module to a second BTCP module located at said selected back-end web server</u>, said initial TCP state associated with said communication session between said client and said first BTCP module, <u>said second BTCP module located below a second TCP module in a second operating system at said selected back-end web server;</u>
>
> f) sending said HTTP request to said selected back-end web server;
>
> g) switching a bottom IP (BIP) module in said front-end node to a forwarding mode, wherein packets, from said client, received at said front-end

node are intercepted by said BIP module and forwarded to said selected back-end web server, said BIP module located below an IP module in said front-end node, said BIP module changing destination IP addresses of said packets to said selected back-end web server and

h) terminating said communication session after said HTTP request has been fully processed. (Emphasis added).

Albert fails to teach all elements of independent claim 11. As described above, *Albert* does not provide a modularized solution, and thus fails to teach at least the modules recited in claim 11.

Specifically, *Albert* fails to teach at least:

A) a first BTCP module located below a first TCP module in a first operating system at a front-end node;

B) parsing an HTTP request to determine which back-end web server contains data in order to process the HTTP request;

C) a bottom IP (BIP) module located below an IP module in the front-end node;

D) handing-off an initial TCP state of said first BTCP module to a second BTCP module located at said selected back-end web server; and

E) a second BTCP module located below a second TCP module in a second operating system at said selected back-end web server.

A. Albert fails to teach a first BTCP module located below a first TCP module in a first operating system at a front-end node

First, independent claim 11 recites "establishing said communication session between said client and said first BTCP module, said first BTCP module located below a first TCP module in a first operating system at said front-end node" (emphasis added). As described above with claim 1, *Albert* fails to teach a first bottom TCP (BTCP) module located below a first TCP module in a first operating system at a front-end node.

B. Albert fails to teach parsing an HTTP request to determine which back-end web server contains data in order to process the HTTP request

Additionally, independent claim 11 further recites "c) receiving a HTTP request from said client at said first BTCP module; d) parsing said HTTP request to determine which back-

end web server, a selected back-end web server, in said plurality of back-end web servers contains said data in order to process said HTTP request, said selected back-end web server not said front-end node" (emphasis added). As described above with claim 1, *Albert* fails to teach parsing a received HTTP request to determine which of a plurality of back-end web servers can process the HTTP request.

C. Albert fails to teach a bottom IP (BIP) module located below an IP module in the front-end node

Further, claim 11 recites "switching a bottom IP (BIP) module in said front-end node to a forwarding mode, wherein packets, from said client, received at said front-end node are intercepted by said BIP module and forwarded to said selected back-end web server, said BIP module located below an IP module in said front-end node, said BIP module changing destination IP addresses of said packets to said selected back-end web server" (emphasis added). As described above with claim 1, *Albert* does not teach modules and specifically does not teach a BIP module located below an IP module at the front-end node (e.g., at the forwarding agent of *Albert*).

D. Albert fails to teach handing-off an initial TCP state of said first BTCP module to a second BTCP module located at said selected back-end web server

Albert fails to teach this further element of claim 11. Instead, the forwarding agent in *Albert* acts as an intermediary to forward communication that it receives to an appropriate back-end server, but the forwarding agent does not hand-off an initial TCP state to a selected back-end server. That is, the forwarding agent forwards packets to the appropriate back-end server, but it does not hand off an initial TCP state to the back-end server.

E. Albert fails to teach a second BTCP module located below a second TCP module in a second operating system at said selected back-end web server

Albert fails to teach this further element of claim 11. That is, *Albert* does not teach a second BTCP module that is located below a second TCP module in a second operating system at a selected back-end web server. *Albert* makes no mention of such a BTCP module in the operating system of a selected back-end server.

Accordingly, *Albert* fails to teach at least the above-identified elements of claim 11, and therefore claim 11 is not anticipated under 35 U.S.C. § 102 by *Albert*. As such, Appellant respectfully requests that the rejection of claim 11 be overturned.

Independent Claim 22

Independent claim 22 recites:

> A communication network for TCP state migration comprising:
> a client;
> a front-end node coupled to said client by said communication network, said front-end node including a front-end bottom TCP (BTCP) module located below a front-end TCP module in a first operating system, and a bottom IP (BIP) module located below an IP module in said first operating system; and
> a plurality of back-end web servers including a selected back-end web server, said plurality of back-end web servers containing content that is partitioned between each of said plurality of back-end web servers, each of said plurality of back-end web servers coupled to said front-end node through said communication network, each of said plurality of back-end web servers including a back-end bottom TCP module located below a back-end TCP module. (Emphasis added).

Albert fails to teach all elements of independent claim 22. As described above, *Albert* does not provide a modularized solution, and thus fails to teach at least the modules recited in claim 22.

Specifically, *Albert* fails to teach at least:

A) a front-end node that includes a front-end bottom TCP (BTCP) module located below a front-end TCP module in a first operating system;

B) a front-end node that further includes a bottom IP (BIP) module located below an IP module in said first operating system; and

C) a back-end web server that includes a back-end bottom TCP module located below a back-end TCP module.

A. Albert fails to teach a front-end node that includes a front-end bottom TCP (BTCP) module located below a front-end TCP module in a first operating system

First, independent claim 22 recites "a front-end node coupled to said client by said communication network, said front-end node including a front-end bottom TCP (BTCP) module located below a front-end TCP module in a first operating system, and a bottom IP (BIP) module located below an IP module in said first operating system" (emphasis added). As described above with claim 1, *Albert* fails to teach a front-end node that includes modules. Particularly, *Albert* does not teach that its forwarding agents include a front-end bottom TCP (BTCP) module located below a front-end TCP module in a first operating system. Thus, *Albert* fails to teach at least this element of claim 22.

B. Albert does not teach a front-end node that further includes a bottom IP (BIP) module located below an IP module in said first operating system

Further, *Albert* does not teach that its forwarding agents include a bottom IP (BIP) module located below an IP module in the first operating system. Thus, *Albert* fails to teach at least this further element of claim 22.

C. Albert does not teach a back-end web server that includes a back-end bottom TCP module located below a back-end TCP module

Further, claim 22 recites "a plurality of back-end web servers including a selected back-end web server, said plurality of back-end web servers containing content that is partitioned between each of said plurality of back-end web servers, each of said plurality of back-end web servers coupled to said front-end node through said communication network, each of said plurality of back-end web servers including a back-end bottom TCP module located below a back-end TCP module" (emphasis added). *Albert* does not teach back-end web servers that include modules, and particularly not back-end web servers that include a back-end bottom TCP module located below a back-end TCP module. Thus, *Albert* fails to teach at least this further element of claim 22.

Accordingly, *Albert* fails to teach at least the above-identified elements of claim 22, and therefore claim 22 is not anticipated under 35 U.S.C. § 102 by *Albert*. As such, Appellant respectfully requests that the rejection of claim 22 be overturned.

Dependent Claims 2, 4, and 8-10

Claims 2, 4, and 8-10 each depend either directly or indirectly from independent claim 1, and thus claims 2, 4, and 8-10 each inherit all elements of claim 1. Therefore, claims 2, 4, and 8-10 are each allowable over *Albert* at least for the reasons discussed above with claim 1. As such, Appellant respectfully requests that the rejection of claims 2, 4 and 8-10 be overturned.

Dependent Claim 3

Dependent claim 3 depends from claim 1 and thus inherits all elements of claim 1. Accordingly, claim 3 is allowable over *Albert* at least for the reasons discussed above with claim 1. Additionally, claim 3 further recites "wherein said back-end web server includes a second BTCP module that is located below a second TCP module in a second operating system" (emphasis added). *Albert* fails to teach this further element of claim 3. As discussed above with claims 11 and 22, *Albert* makes no mention of a second BTCP module that is located below a second TCP module in an operating system of a selected back-end server.

Accordingly, claim 3 is not anticipated under 35 U.S.C. § 102 by *Albert*. As such, Appellant respectfully requests that the rejection of claim 3 be overturned.

Dependent Claim 5

Dependent claim 5 depends from claim 4, which depends from claim 1, and thus claim 5 inherits all elements of claims 1 and 4. Accordingly, claim 5 is allowable over *Albert* at least for the reasons discussed above with claims 1 and 4. Additionally, claim 5 further recites:

> The method as described in Claim 4, wherein said step d) comprises the further steps of:
> sending a SYN packet to said selected back-end web server, said SYN packet intercepted by a second BTCP module, said SYN packet originally sent

> from said client to said front-end node in requesting said communication session, said SYN packet stored at said first BTCP module;
> including an initial sequence number within said SYN packet that enables said second BTCP module to understand a proper TCP state of said first BTCP module in said communication session;
> receiving a SYN/ACK packet from said selected back-end web server, said SYN/ACK packet updated by said second BTCP module to reflect said proper TCP state of said first BTCP module; and
> sending an ACK packet from said first BTCP module to said selected back-end web server, said ACK packet originally sent from said client to said front-end node in establishing said communication session.

As discussed above, *Albert* does not teach the recited first and second BTCP modules. Thus, for at least this reason, *Albert* further fails to teach the steps recited in claim 5. Accordingly, claim 5 is not anticipated under 35 U.S.C. § 102 by *Albert*. As such, Appellant respectfully requests that the rejection of claim 5 be overturned.

Dependent Claim 6

Dependent claim 6 depends from claim 1 and thus inherits all elements of claim 1. Accordingly, claim 6 is allowable over *Albert* at least for the reasons discussed above with claim 1. Additionally, claim 6 further recites:

> The method as described in Claim 1, wherein said method comprises the further step of:
> sending response packets from said selected back-end web server to said client in a communication path that does not include said front-end node by changing headers of said response packets such that it appears that the source of said response packets is said first BTCP in its proper TCP state.

Albert fails to teach this further element of claim 6. That is, *Albert* fails to teach that its back-end servers send a response to a client in a communication path that does not include the front-end node. The Final Office Action asserts that the forwarding agent of *Albert* is the recited front-end node. *Albert* does not teach that its back-end server responds to a client via a communication path that does not include the forwarding agent. Rather, the responsive communication from the back-end servers in *Albert* is sent back through the forwarding agents.

Accordingly, claim 6 is not anticipated under 35 U.S.C. § 102 by *Albert*. As such, Appellant respectfully requests that the rejection of claim 6 be overturned.

Dependent Claim 7

Dependent claim 7 depends from claim 1 and thus inherits all elements of claim 1. Accordingly, claim 7 is allowable over *Albert* at least for the reasons discussed above with claim 1. Additionally, claim 7 further recites:

> The method as described in Claim 1, wherein step g) comprises the further steps of:
> intercepting TCP control packets from a second TCP module located at said selected back-end web server at said second BTCP module;
> sending said TCP control packets to said first BTCP module from said second BTCP module;
> sending said TCP control packets to said client from said first BTCP module; and
> terminating said communication session at said front-end node and said back-end web server.

As discussed above, *Albert* does not teach the recited first and second BTCP modules. Thus, for at least this reason, *Albert* further fails to teach the steps recited in claim 7. Accordingly, claim 7 is not anticipated under 35 U.S.C. § 102 by *Albert*. As such, Appellant respectfully requests that the rejection of claim 7 be overturned.

Dependent Claims 13, 16, and 18-21

Claims 13, 16, and 18-21 each depend either directly or indirectly from independent claim 11, and thus claims 13, 16, and 18-21 each inherit all elements of claim 11. Therefore, claims 13, 16, and 18-21 are each allowable over *Albert* at least for the reasons discussed above with claim 11. As such, Appellant respectfully requests that the rejection of claims 13, 16, and 18-21 be overturned.

Dependent Claim 12

Dependent claim 12 depends from claim 11 and thus inherits all elements of claim 11. Accordingly, claim 12 is allowable over *Albert* at least for the reasons discussed above with claim 11. Additionally, claim 12 further recites:

> 'The method as described in Claim 1, wherein step e) comprises the further steps of:
> e1) storing a SYN packet sent from said client to said front-end

> node, said SYN packet requesting said communication session in step a);
>
> e2) storing an ACK packet sent from said client to said front end node in establishing said communication session;
>
> e3) sending said SYN packet to said selected back-end web server so that it appears that said SYN packet originated from said client;
>
> e4) sending said initial TCP state to said second BTCP module, including said initial sequence number, that enables said second BTCP module to understand a proper TCP state of said first BTCP module for said communication session;
>
> e5) receiving a SYN/ACK packet at said first BTCP module from said second TCP module, said SYN/ACK packet updated by said second BTCP module to reflect said proper TCP state at said first BTCP for said communication session; and
>
> e6) sending said ACK packet to said selected back-end web server to extend said communication session to said selected server.

As discussed above, *Albert* does not teach the recited first and second BTCP modules. Thus, for at least this reason, *Albert* further fails to teach the steps recited in claim 12 that involve those modules. Accordingly, claim 12 is not anticipated under 35 U.S.C. § 102 by *Albert*. As such, Appellant respectfully requests that the rejection of claim 12 be overturned.

Dependent Claim 14

Dependent claim 14 depends from claim 11 and thus inherits all elements of claim 11. Accordingly, claim 14 is allowable over *Albert* at least for the reasons discussed above with claim 11. Additionally, claim 14 further recites:

> The method as described in Claim 11, wherein said method comprises the further step of sending response packets from said back-end web server to said client in a communication path that does not include said front-end node, by changing headers of said response packets such that it appears that the source of said response packets is said front-end node with said proper TCP state.

Albert fails to teach this further element of claim 14. That is, *Albert* fails to teach that its back-end servers send a response to a client in a communication path that does not include the front-end node. The Final Office Action asserts that the forwarding agent of *Albert* is the recited front-end node. *Albert* does not teach that its back-end server responds to a client via a communication path that does not include the forwarding agent. Rather, the responsive communication from the back-end servers in *Albert* is sent back through the forwarding agents.

Accordingly, claim 14 is not anticipated under 35 U.S.C. § 102 by *Albert*. As such, Appellant respectfully requests that the rejection of claim 14 be overturned.

Dependent Claim 15

Dependent claim 15 depends from claim 11 and thus inherits all elements of claim 11. Accordingly, claim 15 is allowable over *Albert* at least for the reasons discussed above with claim 11. Additionally, claim 15 further recites:

> The method as described in Claim 11, wherein step h) comprises the steps of:
> intercepting TCP control packets from said selected back-end web server at said second BTCP module;
> sending said TCP control packets to said first BTCP module from said second BTCP module;
> sending said TCP control packets to said client from said first BTCP module; and
> terminating said communication session at said front-end node and said back-end web server.

As discussed above, *Albert* does not teach the recited first and second BTCP modules. Thus, for at least this reason, *Albert* further fails to teach the steps recited in claim 15 that involve those modules. Accordingly, claim 12 is not anticipated under 35 U.S.C. § 102 by *Albert*. As such, Appellant respectfully requests that the rejection of claim 15 be overturned.

Dependent Claim 17

Dependent claim 17 depends from claim 11 and thus inherits all elements of claim 11. Accordingly, claim 17 is allowable over *Albert* at least for the reasons discussed above with claim 11. Additionally, claim 17 further recites:

> The method as described in Claim 11, wherein said method bypasses the first TCP module.

Albert fails to teach a method that bypasses a TCP module at the front-end node (e.g., forwarding agent of *Albert*). Thus, for at least this reason, claim 17 is not anticipated under 35 U.S.C. § 102 by *Albert*. As such, Appellant respectfully requests that the rejection of claim 17 be overturned.

Dependent Claim 29

Claim 29 depends from independent claim 22 and thus inherits all elements of claim 22. Therefore, claim 29 is allowable over *Albert* at least for the reasons discussed above with claim 22. As such, Appellant respectfully requests that the rejection of claim 29 be overturned.

Dependent Claim 23

Dependent claim 23 depends from claim 22 and thus inherits all elements of claim 22. Accordingly, claim 23 is allowable over *Albert* at least for the reasons discussed above with claim 22. Additionally, claim 23 further recites:

> The communication network as described in Claim 22, wherein said front-end BTCP module establishes a communication session with said client for the transfer of data contained within said content to said client.

As discussed above, *Albert* does not teach the recited front-end BTCP module, and thus fails to teach establishing a communication session with a client as recited by this element of claim 23. Accordingly, claim 23 is not anticipated under 35 U.S.C. § 102 by *Albert*. As such, Appellant respectfully requests that the rejection of claim 23 be overturned.

Dependent Claim 24

Dependent claim 24 depends from claim 23, which depends from claim 22, and thus claim 24 inherits all elements of claims 22 and 23. Accordingly, claim 24 is allowable over *Albert* at least for the reasons discussed above with claims 22-23. Additionally, claim 24 further recites:

> The communication network as described in Claim 23, wherein said front-end BTCP module parses a HTTP request from said client in order to determine which of said plurality of back-end web servers, a selected back-end web server, contains said data in order to process said HTTP request.

As discussed above with claims 1 and 11, *Albert* does not teach parsing a HTTP request from a client in order to determine which of the plurality of back-end web servers

contains the data. Accordingly, claim 24 is not anticipated under 35 U.S.C. § 102 by *Albert*. As such, Appellant respectfully requests that the rejection of claim 24 be overturned.

Dependent Claim 25

Dependent claim 25 depends from claim 23, which depends from claim 22, and thus claim 25 inherits all elements of claims 22 and 23. Accordingly, claim 25 is allowable over *Albert* at least for the reasons discussed above with claims 22-23. Additionally, claim 25 further recites:

> The communication network as described in Claim 23, wherein said front-end BTCP module extends said communication session to said selected back-end web server by handing-off an initial TCP state of said front-end BTCP module to a second BTCP module located at said selected back-end web server, said initial TCP state associated with a proper TCP state for said front-end BTCP module in said communication session, said front-end BTCP module further forwarding packets, including said HTTP request, from said client after successfully handing-off said initial TCP state.

As discussed above, *Albert* does not teach the recited front-end BTCP module. *Albert* further fails to teach the recited second BTCP module located at the selected back-end web server. Thus, claim 25 is not anticipated under 35 U.S.C. § 102 by *Albert*. As such, Appellant respectfully requests that the rejection of claim 25 be overturned.

Dependent Claim 26

Dependent claim 26 depends from claim 25, which depends from claim 23 which depends from claim 22, and thus claim 26 inherits all elements of claims 22-23 and 25. Accordingly, claim 26 is allowable over *Albert* at least for the reasons discussed above with claims 22-23 and 25. Additionally, claim 26 further recites:

> The communication network as described in Claim 25, wherein said second BTCP module understands said proper TCP state of said front-end BTCP module in said communication session and modifies headers in response packets from said selected back-end web server to reflect said proper TCP state.

As discussed above, *Albert* does not teach the recited second BTCP module or the front-end BTCP module. *Albert* further fails to teach such a second BTCP that understands

the proper TCP state of the front-end BTCP module and modifies headers in response packets as recited in claim 26. Thus, claim 26 is not anticipated under 35 U.S.C. § 102 by *Albert*. As such, Appellant respectfully requests that the rejection of claim 26 be overturned.

Dependent Claim 27

Dependent claim 27 depends from claim 25, which depends from claim 23 which depends from claim 22, and thus claim 27 inherits all elements of claims 22-23 and 25. Accordingly, claim 27 is allowable over *Albert* at least for the reasons discussed above with claims 22-23 and 25. Additionally, claim 27 further recites:

> The communication network as described in Claim 25, wherein said BIP module changes a destination address in forwarding said packets from said client.

As discussed above, *Albert* does not teach the recited BIP module. *Albert* further fails to teach such a BIP module that changes a destination address in forwarding packets from the client, as recited in claim 27. Thus, claim 27 is not anticipated under 35 U.S.C. § 102 by *Albert*. As such, Appellant respectfully requests that the rejection of claim 27 be overturned.

Dependent Claim 28

Dependent claim 28 depends from claim 26, which depends from claim 25 which depends from claim 23 which depends from claim 22, and thus claim 28 inherits all elements of claims 22-23 and 25-26. Accordingly, claim 28 is allowable over *Albert* at least for the reasons discussed above with claims 22-23 and 25-26. Additionally, claim 28 further recites:

> The communication network, as described in Claim 26, wherein said second BTCP module located at said selected back-end web server sends said response packets from said selected back-end web server to said client in a communication path that does not include said front-end node by changing headers of said response packets such that it appears the source of said response packets is said front-end node.

Albert fails to teach this further element of claim 28. That is, *Albert* fails to teach that its back-end servers send a response to a client in a communication path that does not include the front-end node. The Final Office Action asserts that the forwarding agent of *Albert* is the recited front-end node. *Albert* does not teach that its back-end server responds to a client via

a communication path that does not include the forwarding agent. Rather, the responsive communication from the back-end servers in *Albert* is sent back through the forwarding agents.

Accordingly, claim 28 is not anticipated under 35 U.S.C. § 102 by *Albert*. As such, Appellant respectfully requests that the rejection of claim 28 be overturned.

VIII. CLAIMS

A copy of the claims involved in the present appeal is attached hereto as Appendix A.

IX. EVIDENCE

As noted in Appendix B hereto, no evidence pursuant to §§ 1.130, 1.131, or 1.132 or entered by or relied upon by the examiner is being submitted.

X. RELATED PROCEEDINGS

The related appeal identified in Section II is listed in Appendix C attached hereto. No decision has been received for the related appeal, and thus no copy of a decision is provided.

I hereby certify that this correspondence is being deposited with the United States Postal Service as Express Mail, Label No. EV 4568257578US in an envelope addressed to: M/S Appeal Brief Patents, Commissioner for Patents, Alexandria, VA 22313.

Date of Deposit: January 5, 2006

Typed Name: Gail L. Miller

Signature: Gail L. Miller

Respectfully submitted,

By: Jody C. Bishop

Jody C. Bishop
Attorney/Agent for Applicant(s)
Reg. No. 44,034
Date: January 5, 2006
Telephone No. (214) 855-8007

APPENDIX A

Claims Involved in the Appeal of Application Serial No. 09/880,632

1. In a communication network, a method of TCP state migration comprising the steps of:

a) establishing a communication session between a client and a front-end node at a first bottom TCP (BTCP) module located below a first TCP module in a first operating system at said front-end node, said front-end node accessing a plurality of back-end web servers forming a web server cluster that contains content;

b) receiving a HTTP request from said client at said first BTCP module;

c) parsing said HTTP request to determine which back-end web server, a selected back-end web server, in said plurality of back-end web servers can process said HTTP request, said selected back-end web server not said front-end node;

d) extending said communication session to said selected back-end web server by handing-off an initial TCP state of said first BTCP module to said selected back-end web server;

e) sending said HTTP request to said selected back-end web server;

f) switching a bottom IP (BIP) module at said front-end node to a forwarding mode, wherein packets received at said BIP module from said client are forwarded to said selected back-end web server, said BIP module located below an IP module at said front-end node; and

g) terminating said communication session at said front-end node after said HTTP request is fully processed.

2. The method as described in Claim 1, wherein said content is partially replicated between each of said plurality of back-end web servers.

3. The method as described in Claim 1, wherein said back-end web server includes a second BTCP module that is located below a second TCP module in a second operating system.

4. The method as described in Claim 1, wherein said initial TCP state is associated with said communication session, said communication session established for the transfer of data contained within said content to said client.

5. The method as described in Claim 4, wherein said step d) comprises the further steps of:

sending a SYN packet to said selected back-end web server, said SYN packet intercepted by a second BTCP module, said SYN packet originally sent from said client to said front-end node in requesting said communication session, said SYN packet stored at said first BTCP module;

including an initial sequence number within said SYN packet that enables said second BTCP module to understand a proper TCP state of said first BTCP module in said communication session;

receiving a SYN/ACK packet from said selected back-end web server, said SYN/ACK packet updated by said second BTCP module to reflect said proper TCP state of said first BTCP module; and

sending an ACK packet from said first BTCP module to said selected back-end web server, said ACK packet originally sent from said client to said front-end node in establishing said communication session.

6. The method as described in Claim 1, wherein said method comprises the further step of:

sending response packets from said selected back-end web server to said client in a communication path that does not include said front-end node by changing headers of said response packets such that it appears that the source of said response packets is said first BTCP in its proper TCP state.

7. The method as described in Claim 1, wherein step g) comprises the further steps of:

intercepting TCP control packets from a second TCP module located at said selected back-end web server at said second BTCP module;

sending said TCP control packets to said first BTCP module from said second BTCP module;

sending said TCP control packets to said client from said first BTCP module; and

terminating said communication session at said front-end node and said back-end web server.

8. The method as described in Claim 1, wherein said front-end node and said plurality of back-end web servers comprise a web site, said front-end node providing a virtual IP address for said web site.

9. The method as described in claim 8, wherein said front-end node, and said plurality of back-end web servers are coupled together by a local area network.

10. The method as described in Claim 8, wherein said front-end node and said plurality of back-end web servers are coupled together by a wide area network,.

11. In a communication network, a method of TCP state migration comprising the steps of:

a) receiving a request from a client for establishing a communication session at a first bottom TCP (BTCP) module located at a front-end node, said front-end node accessing a plurality of back-end web servers containing content, wherein said content is partially replicated between each of said plurality of back-end web servers, said communication session established for the transfer of data contained within said content to said client;

b) establishing said communication session between said client and said first BTCP module, said first BTCP module located below a first TCP module in a first operating system at said front-end node;

c) receiving a HTTP request from said client at said first BTCP module;

d) parsing said HTTP request to determine which back-end web server, a selected back-end web server, in said plurality of back-end web servers contains said data in order to process said HTTP request, said selected back-end web server not said front-end node;

e) extending said communication session to said selected back-end web server by handing-off an initial TCP state of said first BTCP module to a second BTCP module located at said selected back-end web server, said initial TCP state associated with said communication session between said client and said first BTCP module, said second BTCP module located below a second TCP module in a second operating system at said selected back-end web server;

f) sending said HTTP request to said selected back-end web server;

g) switching a bottom IP (BIP) module in said front-end node to a forwarding mode, wherein packets, from said client, received at said front-end node are intercepted by said BIP module and forwarded to said selected back-end web server, said BIP module located below an IP module in said front-end node, said BIP module changing destination IP addresses of said packets to said selected back-end web server and

h) terminating said communication session after said HTTP request has been fully processed.

12. The method as described in Claim 1, wherein step e) comprises the further steps of:

e1) storing a SYN packet sent from said client to said front-end node, said SYN packet requesting said communication session in step a);

e2) storing an ACK packet sent from said client to said front end node in establishing said communication session;

e3) sending said SYN packet to said selected back-end web server so that it appears that said SYN packet originated from said client;

e4) sending said initial TCP state to said second BTCP module, including said initial sequence number, that enables said second BTCP module to understand a proper TCP state of said first BTCP module for said communication session;

e5) receiving a SYN/ACK packet at said first BTCP module from said second TCP module, said SYN/ACK packet updated by said second BTCP module to reflect said proper TCP state at said first BTCP for said communication session; and

e6) sending said ACK packet to said selected back-end web server to extend said communication session to said selected server.

13. The method as described in Claim 12, wherein step e4) includes the further step of including said initial sequence number in said SYN packet.

14. The method as described in Claim 11, wherein said method comprises the further step of sending response packets from said back-end web server to said client in a communication path that does not include said front-end node, by changing headers of said response packets such that it appears that the source of said response packets is said front-end node with said proper TCP state.

15. The method as described in Claim 11, wherein step h) comprises the steps of:

intercepting TCP control packets from said selected back-end web server at said second BTCP module;

sending said TCP control packets to said first BTCP module from said second BTCP module;

sending said TCP control packets to said client from said first BTCP module; and

terminating said communication session at said front-end node and said back-end web server.

16. The method as described in Claim 15, wherein said TCP control packets include a RST flag and a FIN flag.

17. The method as described in Claim 11, wherein said method bypasses the first TCP module.

18. The method as described in Claim 11, wherein said front-end node, and said plurality of back-end web servers comprise a web site, said front-end node providing a virtual IP address for said web site.

19. The method as described in claim 18, wherein said front-end node, and said plurality of back-end web servers are coupled together by a local area network.

20. The method as described in Claim 18, wherein said front-end node and said plurality of back-end web servers are coupled together by a wide area network.

21. The method as described in Claim 11, wherein said content is partitioned between each of said plurality of back-end web servers.

22. A communication network for TCP state migration comprising:

a client;

a front-end node coupled to said client by said communication network, said front-end node including a front-end bottom TCP (BTCP) module located below a front-end TCP module in a first operating system, and a bottom IP (BIP) module located below an IP module in said first operating system; and

a plurality of back-end web servers including a selected back-end web server, said plurality of back-end web servers containing content that is partitioned between each of said plurality of back-end web servers, each of said plurality of back-end web servers coupled to said front-end node through said communication network, each of said plurality of back-end web servers including a back-end bottom TCP module located below a back-end TCP module.

23. The communication network as described in Claim 22, wherein said front-end BTCP module establishes a communication session with said client for the transfer of data contained within said content to said client.

24. The communication network as described in Claim 23, wherein said front-end BTCP module parses a HTTP request from said client in order to determine which of said plurality of back-end web servers, a selected back-end web server, contains said data in order to process said HTTP request.

25. The communication network as described in Claim 23, wherein said front-end BTCP module extends said communication session to said selected back-end web server by handing-off an initial TCP state of said front-end BTCP module to a second BTCP module located at said selected back-end web server, said initial TCP state associated with a proper TCP state for said front-end BTCP module in said communication session, said front-end BTCP module further forwarding packets, including said HTTP request, from said client after successfully handing-off said initial TCP state.

26. The communication network as described in Claim 25, wherein said second BTCP module understands said proper TCP state of said front-end BTCP module in said communication session and modifies headers in response packets from said selected back-end web server to reflect said proper TCP state.

27. The communication network as described in Claim 25, wherein said BIP module changes a destination address in forwarding said packets from said client.

28. The communication network, as described in Claim 26, wherein said second BTCP module located at said selected back-end web server sends said response packets from said selected back-end web server to said client in a communication path that does not include said front-end node by changing headers of said response packets such that it appears the source of said response packets is said front-end node.

29. The communication network as described in Claim 22 wherein said content is partially replicated between each of said plurality of back-end web servers.

APPENDIX B

Evidence

None.

APPENDIX C

Related Proceedings

A currently pending appeal before the Board of co-pending U.S. Patent Application Serial No. 09/880,631, and particularly the Board's interpretation of *Albert*, may affect, be affected by, or have a bearing on the Board's decision in this appeal.

Application No. (if known): 09/880,632 Attorney Docket No10012351-1

Certificate of Express Mailing Under 37 CFR 1.10

I hereby certify that this correspondence is being deposited with the United States Postal Service as Express Mail, Airbill No. EV 568257578US in an envelope addressed to:

MS Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

on January 5, 2006
Date

Gail Miller
Signature

Gail Miller
Typed or printed name of person signing Certificate

Registration Number, if applicable

(214) 855-8379
Telephone Number

Note: Each paper must have its own certificate of mailing, or this certificate must identify each submitted paper.

2 Return Postcards
Transmittal – 1 page
Corrected Appeal Brief – 39 pages
Certificate of Mail – 1 page